Types de variables :

Pseudo-code	Python
Entier	int
Réel	float
Texte	str
Logique	bool

Opérateurs arithmétiques :

Opérateurs arithmétiques	Python
+	+
-	-
*	*
/	/
DIV	//
MOD	%

Opérateurs de comparaisons :

Opérateurs de comparaison	Python
Égal	==
Différent	!=
Strictement plus grand	>
Strictement plus petit	<
Plus grand ou égal	>=
Plus petit ou égal	<=

Opérateurs logiques :

Opérateurs logiques	Python
NON	not
ET	and
OU	or

Opération Écrire()

```
prix = 0.99
print("Prix : ", prix)
```

Opération Lire()

En Python, lorsque nous lisons ce qui est saisi par l'utilisateur, nous récupérons une chaine de caractères. Si nous désirons obtenir une valeur de type différent, nous devons la convertir.

```
nombre = int(input("Entrez un nombre"))
message = input("Entrez un message"))
```

"nombre" étant un entier et "message" étant une chaine de caractères

Effacer l'écran

```
import os
os.system('cls')
```

Déclaration et affectation de variables :

```
age = 10
temperature = 15.6
sexe = 'M'
message = "Bonsoir"
estVide = True
```

Structures conditionnelles :

SI..SINON

```
if condition :
    # Si

else :
    # Sinon
```

SI..SINON SI..SINON

```
if condition :
    # Si

elif condition :
    # Sinon si

else :
    # Sinon
```

Générer un nombre aléatoire

```
import random
nbr = random.randint(0, 100)
# ou
nbr = int(random.random() * 100)
# ou
nbr = random.randrange(0,11,1)
```

La ligne suivante nous permet d'obtenir un nombre aléatoire compris entre 0 et 100.

Structures itératives

TANT QUE .. FAIRE

```
while condition :
  #instructions à répéter
```

Les boucles FAIRE .. TANT QUE et FAIRE .. JUSQU'À n'existent pas en Python

Raccourcis

Incrémentation

L'opération i = i + 1 peut être remplacée par i+=1

Décrémentation

L'opération i = i - 1 peut être remplacée par i-=1

Les tableaux

Déclaration

Nous déclarons un tableau de la manière suivante en Python

```
nomTab = []
```

Nous pouvons également initialiser un tableau à sa déclaration :

```
nomTab = [ 1, 5, 3, 5, 2, 6, 5, 8 ]
```

La ligne ci-dessus nous permet de déclarer un tableau contenant 8 éléments dont les valeurs sont exprimées entre les crochets et séparées par des virgules.

Opérateur d'accès [ ]

Les crochets nous permettent d'accéder à un élément donné.

```
nomTab[0] = 10
```

La ligne ci-dessus va affecter la valeur 10 dans la première case du tableau.

Boucle POUR

La boucle pour nous permet de parcourir le tableau.

```
for element in nomTab :
    # instructions
    print(element)
```

Les sous-programmes

Les procédures et les fonctions

En Python (comme dans beaucoup d'autres langages), il n'existe pas de procédure à proprement parler.

```
def procedure (parametre1, parametre2) :
    # instructions
```

L'encadré ci-dessus illustre comment déclarer une procédure. Dans ce cas, il s'agit plus précisément d'une fonction qui ne renvoie pas de valeur, mais qui reçoit deux paramètres.

Lorsque nous désirons faire une fonction qui renvoie une valeur, nous devons préciser le type de la valeur qui sera renvoyée. De plus, l'envoi de cette valeur se fait à l'aide du mot-clé "return".

```
def fonction(parametre1, parametre2) :
    # instructions
    return "valeur de retour de type string"
```